

Time Synchronization of New Devices in Ad-hoc Multimedia Networks

Manikanden Balakrishnan^{†§} and Larry Taylor[§]

[§]Staccato Communications, Inc.
San Diego, CA 92121

[†]New Mexico State University
Las Cruces, NM 88003

mabalakr@nmsu.edu, larry.taylor@staccatocommunications.com

Abstract – The current WiMedia standards for wireless networks formulate a distributed Medium Access Control (MAC) protocol that incorporates a beaconing mechanism for efficient time coordination of devices in a communication group. The protocol suggests methods whereby devices locally correct relative clock drifts to maintain synchronization with other devices within radio range, without the help of any common infrastructure (such as GPS).

However, the suggested mechanism works only in steady state conditions i.e. maintains synchronization in an already coordinated group. There is an inherent problem during an initial transient period when new devices try to align with an existing group for the first time, which, if not corrected, might not lead to a steady state.

This paper presents a simple, yet important, addition to WiMedia synchronization that precisely time-aligns new devices. The existing problem and the performance of our solution are substantiated through ns-2 simulations.

Keywords – Wireless MAC protocols, time synchronization, peer-to-peer networking, ns-2

I. INTRODUCTION

Wireless networks for Consumer Electronics (CE) continue to generate significant interest in both the research and product development communities. High performance wireless technology for cable replacement and Wireless Personal Area Networking (WPAN) applications is progressing rapidly, particularly with recent advances in UltraWideBand [1] technology. Self-organizing, distributed, peer-to-peer ad-hoc [2] style networking is preferred for CE multimedia applications for simplicity of deployment in consumer environments.

WiMedia [3, 4, 5] is a recent set of standard specifications for high data rate WPAN, and is suited to home multimedia applications. The protocols are designed to suit ad-hoc architectures to provide good scalability and robustness to network dynamics (such as adding new devices, mobility etc.), essential for home wireless networks. Typical applications include communication architectures for mobile devices (PDA, digital cameras, cell-phones) as well as static devices (set-top box, computers, peripherals). The scope of this paper is limited to the discussion of the WiMedia Medium Access Control (MAC) [5] protocol, where time synchronization is addressed.

For the purposes of this paper, it is sufficient to know that the WiMedia MAC layer is serviced by a multi-band OFDM [3, 4] physical layer (PHY) that enables data rates up to 480

Mbps at up to 10 meters range. Serviced by such a high speed channel, the MAC layer must operate in a time frame of the order of micro-seconds (μ s).

A. WiMedia MAC

The WiMedia standards define a distributed MAC protocol for infrastructure-less wireless networks (ad-hoc nets [2]). This section provides a brief conceptual review for background knowledge, as required for this paper's scope. The complete MAC description can be found in [5].

The standard defines a basic timing structure for channel access, called the superframe (SF), which is a period of 65.536 milliseconds (ms) divided into 256 Medium Access Slots (MASs) of 256 μ s each. The SF is a periodically repeating structure to which the devices in a group synchronize and within which the devices' frame transmissions are multiplexed. The SF is divided into two parts:

Beacon Period (BP): Reserved exclusively for the exchange of beacons¹ using a collision resolution mechanism to provide a contention-free, slotted channel access mechanism. Beacons identify and coordinate devices in mutual communications range. The Beacon Protocol [5] establishes all the rules concerning instantiation, maintenance, and modification of a BP.

Data Period: Two different channel access mechanisms operate in the data period. A Distributed Reservation Protocol (DRP), supported by beacon exchange, allows non-preemptive channel reservations with a variety of access rules to be established. Alternatively, a random access channel mechanism derived from 802.11e [6] is provided by the Prioritized Contention Access (PCA) protocol.

In this paper, on time synchronization, we concentrate only on the beacon protocol which provides the basic network timing and incorporates mechanisms to maintain device synchronization.

II. BEACON PROTOCOL

As shown in Figure 1, a SF begins at the Beacon Period Start Time (BPST) which is the beginning of the 1st beacon slot in the variable length BP. All active devices transmit a beacon frame in a beacon slot.

¹ Control frames, exchanged between nodes within radio range

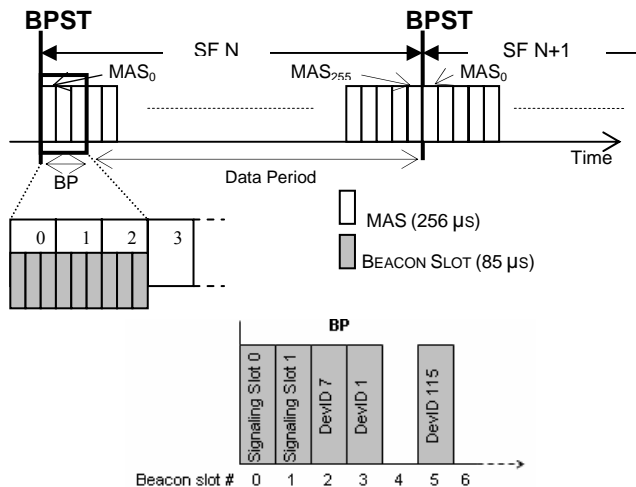


Figure 1. WiMedia SF Structure and BP

The beacon protocol includes algorithms to ensure that each device selects a different beacon slot and that the beacon slot occupancy is maintained in as compact a form as possible, growing and shrinking as the number of devices within range varies. We will note here, but leave it to the reader to explore separately, that the WiMedia MAC is a distributed design and each device operates from its perspective of the network topology. Control algorithms ensure that no two beacon frames are transmitted at the same time within a two hop neighborhood of any device.

Devices must listen for beacons from the BPST up to a beacon slot declared by the device in its beacon and known as the BPLength. Each device maintains the BPLength to cover the beacon slots of its neighbors. The initial two slots are reserved for signaling the presence of new devices whose selected beacon slot may be beyond a device’s declared BPLength. A new device repeats its beacon in a signaling slot until its neighbors extend their BPLengths to accommodate the new device. BPST and BPLength together define a BP, and all devices maintain up-to-date BP information in their local record. The set of devices exchanging beacons comprise a Beacon Group (BG).

Periodically, a device will listen in its own beacon slot to detect beacon collisions which are resolved by selecting a new beacon slot. Should gaps appear in the BP (for example if a device moves out of range or stops operating) they will be filled by devices that are using higher beacon slots to keep the BP compacted.

The beacon protocol is a signaling and control protocol, and beacons carry a variety of network management information in Beacon Parameters and Information Elements (IEs). Beacon Parameters include the device’s unique address and the beacon slot in which the beacon was transmitted. This key piece of information is used to determine the BPST since beacon slots are fixed length, defined to be 85μs long [5].

IEs carry information about the device’s neighborhood (BP Occupancy IE) as well as declarations of, and negotiation for, DRP data reservations (DRP IEs). Since beacons carry a description of the device’s neighborhood and declared reser-

vations (which must be honored) the beacon protocol provides support for spatial re-use as devices beyond two-hop range may re-use the same beacon slots. Mechanisms are also defined for merging of BPs for mobility cases as well as maintenance of BP information for devices in power saving states.

Hereafter, this paper concentrates on the time coordination of devices achieved by the beacon exchanges. The complete Beacon Protocol can be referred from the standard [5].

III. DEVICE SYNCHRONIZATION

Synchronization Definition: Synchronization is defined from the perspective of the periodic SF structure. Two devices are said to be synchronized when they share a common BPST every repeating cycle. The WiMedia standard requires that the BPSTs be aligned to at least 1μs resolution².

This section describes the WiMedia MAC protocol approach to BPST correction for maintaining synchronization.

A. Clock Drift

BPST correction is required to maintain synchronization because each device experiences clock drift. The current WiMedia standard requires that compliant devices maintain a clock accurate to at least 20 parts/million (ppm). Clock drift is a *fixed* ppm error in the devices’ clocking rate. So the worst case clock drift between two nodes would be 40 ppm (one device is slower by 20 ppm and another device is faster by 20 ppm from the absolute clock rate).

Clock inaccuracy is a function of the time elapsed since a synchronization event, and figure 2 depicts the concept.

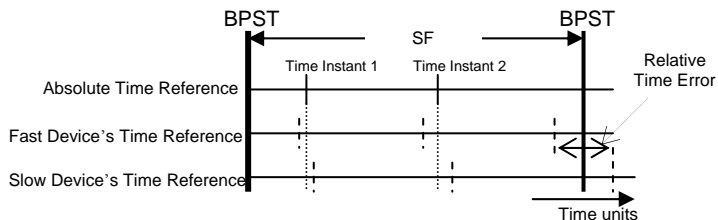


Figure 2. Relative Timing Inaccuracy between Two Nodes

Assume the devices are synchronized at the beginning of the SF (common BPST). Here, timing implies how devices time their operation and not their actual local clock times. *Two devices are said to be perfectly time aligned when they experience the same event (e.g. start of a SF) at the same instant.* Their precise local clock times are irrelevant.

A fast device will schedule events ahead of nominal event time according to an absolute time reference, and a slow device will schedule them trailing the same event scheduled at the nominal clock rate. The error with respect to the absolute timing increases with elapsed time from the BPST, according to the following expression;

$$\text{Relative Time Error (sec)} = \text{Time elapsed from BPST}^3 \\ * \text{Clock Drift between the nodes (ppm)}$$

² Since MAC operates at μs time frame, it is required that synchronization be maintained at least at that resolution for proper channel multiplexing

³ BPST is the previous synchronized point

The maximum relative time error (sec) between two nodes over a SF period would be $65.536\text{ms} * 40\text{ppm} \approx 2.6 \mu\text{s}$.

B. Beacons for Device Synchronization

All devices in a BG align their BPSTs, and all timings are derived from the current BPST (e.g. time to transmit beacon = BPST + beacon slot number * $85\mu\text{s}$). Relative timing errors with neighbors' are computed during the BP and corrected before the end of a SF, ensuring that the start of next SF is always aligned. Devices adjust their BPSTs to maintain synchronization with the *slowest device* in a group. The slowest member of the group would be the last one to finish its SF, and so other devices in the group should delay the start of their next SF and thus align with the slowest device.

Figure 3 explains the concept of BPST correction with a simple example. Assume A's clock is accurate, and B's clock is slow, with respect to an absolute clock rate for simplicity in explanation. Assume also that A and B are aligned at the start of SF N. The key instants are numbered and their corresponding explanation follows:

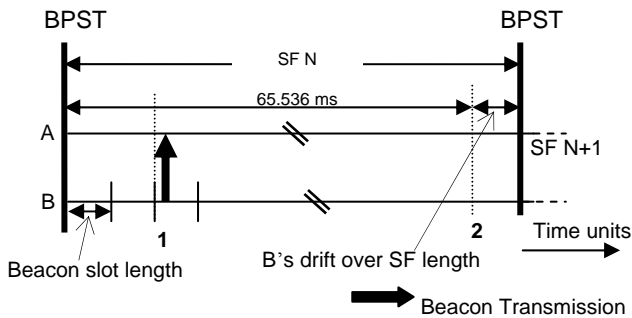


Figure 3. Concept of Maintaining Synchronization

At 1)

Assume it is B's beacon slot. B's beacon transmission would be late relative to A, since B's clock is slow and would reach the scheduled point later than the absolute clock. When A receives the beacon it computes;

$$\begin{aligned} \text{The time at which A expects a beacon from B;} \\ \text{Expected Time} &= \text{BPST} + 85\mu\text{s} * \text{B's Beacon slot number} \\ \text{Actual Time} &= \text{Local time at A when the first symbol of} \\ &\quad \text{B's beacon was received} \end{aligned}$$

$$\text{Relative Time Error (sec)} = \text{Actual Time} - \text{Expected Time}$$

This is the relative timing offset between A and B over an elapsed time of: Expected Time - BPST. Also the value is positive (Actual Time > Expected Time), since B sends the beacon late. At this point, A knows that B is slow.

$$\begin{aligned} \text{Relative Time Error (ppm)}^4 = \\ \text{Relative Time Error (sec)} / (\text{Expected Time} - \text{BPST}) \end{aligned}$$

The ppm error could be mapped to the error (sec) over a SF period.

$$\begin{aligned} \text{Relative Time Error (sec) over a SF} = \\ \text{Relative Time Error (ppm)} * 65.536 \text{ ms} \end{aligned}$$

⁴ Relative Time Error (ppm) is the clock drift (ppm) between the devices. It is the primary measure needed to compute time error (sec) over any interval.

This is the offset amount by which B's clocking instants will lag behind A's over A's SF period. A stores this offset value.

At 2)

A has the normal SF end event, and delays the start of SF N+1 by the stored offset value. After the delay, A aligns with B (who finishes the SF with its local clock error), and both nodes start SF N+1. This algorithm is repeated every cycle.

Key Points

B will do the same computations when receiving A's beacon. But the *Relative Time Error* would be a negative value, and so B will know it is the slower of the two and will ignore any further computations.

In the case of multiple devices, exactly the same computations are performed on every beacon reception. But the devices remember the *highest positive offset* value and they delay the start of next SF by that amount pro rata to the SF duration. This makes sure that the group aligns with the slowest neighbor.

How does this work? This synchronization procedure works on the basis that, nodes in a group share a common reference point, the BPST. Since all timings are derived from the BPST, relative drifts can be locally corrected based on the beacon reception times and the BPST reference point, before the start of the next SF. This is a distributed method of device synchronization.

IV. SYNCHRONIZATION OF NEW DEVICES

The procedure described in the previous section maintains synchronization among devices that have reached steady state (already coordinated with a common BPST). This section explains the inherent deficiency of this algorithm to time-align new devices and presents a solution to this deficiency.

A. Problem Description

Statement: A new device, upon beacon reception for the first time, cannot compute relative time error (slow or fast) with respect to the transmitter, as per the current WiMedia synchronization procedure, since there is not yet a common reference BPST. The estimated BPST correction might not enable time alignment as explained using figure 4.

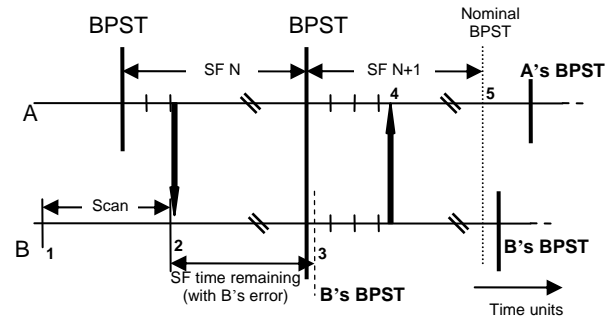


Figure 4. New Device Synchronization Problem

Again assume that A is accurate to the absolute clock and B is slower. A has already instantiated a "single" node group (and is therefore aligned) and B is a new device. The descrip-

tion corresponding to the numbered time instants follows:

1) B switches ON and starts scanning for at least one SF as per the beacon protocol

2) B receives A's beacon. Since B has no previous reference BPST it cannot compute relative time error with A (cannot estimate *Expected Time*), but tries to align with the A's SF using A's implied BPST. B schedules the start of next SF based on A's beacon slot number (declared in A's beacon) as,

$$\text{Remaining time in SF } N = 65.536 \text{ ms} - \text{A's beacon slot number} * 85\mu\text{s}$$

This time has not yet been corrected for the relative drift between A and B.

3) B ends SF N, but the BPSTs are not aligned

4) A receives B's beacon and performs the computations as described in section III.B.

At this instant, *Actual Time - Expected Time* is the error over approximately a SF length, since there has been no synchronization event from the time B instantiated. Due to the distributed nature of the protocol, A does not know that B is new, and wrongly assumes that the *Relative Time Error (sec)* is over a time elapsed from its BPST to this instant (A thinks B was aligned at BPST of N+1).

This wrong assumption results in erroneous further computations. When A maps this offset over to a SF length, it would be a significantly high (and wrong) value⁵

5) A's normal SF end event occurs, and A delays the end of SF by the wrongly computed huge offset value.

B would finish normally with its local clock error, and the BPSTs would be considerably misaligned. Further in SF N+2, B will overcompensate, since it will assume A is considerably slower. This error might keep oscillating and both the nodes could over-compensate for each other alternately.

Underlying reason for this problem and scope

The new device, while setting its first BPST, will not have any offset corrections possible due to lack of reference points when instantiating. This initial error will be wrongly assumed to be over a small elapsed time, and will be wrongly mapped to over a SF length. *This error only occurs when a new device tries to align with an established group, but, if not corrected, might never settle.*

B. Solution Description (New-Device Alignment Correction)

A simple addition to the current WiMedia MAC synchronization procedure would correct this problem. New devices mark the first beacon reception time instant as their reference. Upon next beacon reception from the *same sender* (in next SF), they compute the relative time error using the marked reference. The computed relative offset with sender would be over a SF time, but could be mapped to the ppm error, which is the measure needed.

In short, the 1st beacon reception point serves as the reference and the 2nd beacon reception instant computes the rela-

⁵ The current WiMedia standard limits the maximum offset correction in a SF to 4μs; details explained in [5]

tive time error. Figure 5 uses the same example to explain the solution (matched numbering).

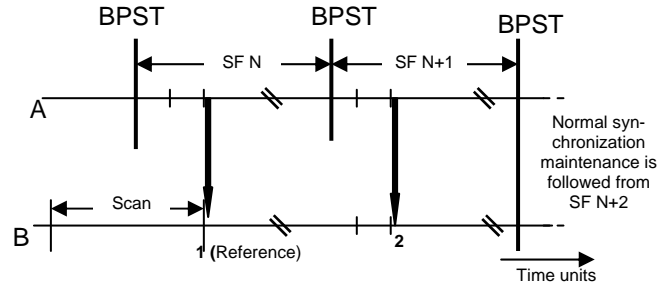


Figure 5. New Device Synchronization: Solution

1) B marks the first beacon reception instant as the reference time (based on its local clock).

2) 2nd beacon reception from same neighbor, A. B computes;

The time at which B expects a beacon from A,

$$\text{Expected Time} = \text{Reference Time} + 65.536\text{ms}^6$$

$$\text{Actual Time} = \text{Local time at B when the first symbol of A's beacon was received}$$

$$\text{Relative Time Error (sec)} = \text{Actual Time} - \text{Expected Time}$$

This value is over SF length, and can be positive or negative. In this case, since B is slower, the value would be negative.

Relative Time Error (in ppm)

$$= \text{Relative Time Error (in sec)} / (65.536\text{ms})$$

At 2, B schedules a SF end event after:

$$(\text{Remaining time in SF } N+1) + \alpha + \beta + \gamma$$

α = Relative error during remaining time in this SF

β = Relative error over SF (from 1st beacon reception to this point)

γ = Relative error during the period from BPST to A's slot (drift before 1st beacon reception in SF N).

α , β and γ are the drift corrections (sec)⁷, and can be directly computed from the estimated ppm error. For example,

$$\alpha = \text{Relative Time Error (in ppm)} * \text{Remaining time in SF } N+1$$

At the SF end event instant B aligns with A.

Notes

This alignment correction procedure consumes two SFs after the time of new device instantiation, but assures perfect time synchronization. After the new device coordinates itself with the existing SF structure, the normal process in section III.B is followed to maintain synchronization.

This algorithm inductively converges to multiple node cases. If the group size is more than two, A will BPST correct for the slowest device at the end of SF N. Since B is referencing A, it's computed error will be relative with the slowest device i.e., A's 2nd beacon transmission would be influenced by the delay of the slowest device in the group, which will automatically reflect in B's computations.

⁶Normally, A sends a beacon every SF at the same beacon slot. In case of beacon losses or slot modifications [5], the formula would incorporate additional terms for robustness. Those optimizations are ignored in this discussion.

⁷ α , β and γ can be negative values too. In this case, since B is slower, the corrections push B's SF end event earlier so that it aligns with A.

V. IMPLEMENTATION

This section provides experimental results to verify the correctness of our new-device synchronization method.

A. NS-2 WiMedia MAC model

Our research group has built a concrete simulation model of the WiMedia standard for the purpose of research and development. Simulations provide an ideal platform for fast and repeated experimentations on the protocol technologies under various network configurations. The MAC model is developed in the Network Simulator-2 (ns-2) environment [7]. ns-2 is a popular discrete event simulator extensively used in the wireless academic research community. Figure 6 shows the structure of the WiMedia MAC stack embedded into the ns-2 environment.

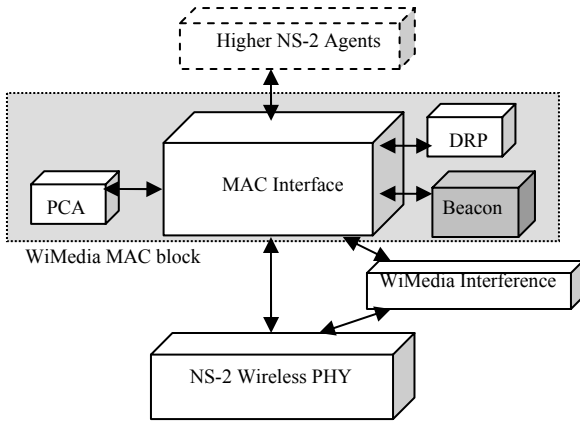


Figure 6. WiMedia Stack in ns-2

MAC interface provides the WiMedia embedding functionalities like instantiating MAC objects, functional switching between DRP, Beacon and PCA modules, functional switching across layers, event scheduling, clock drift implementation etc. The Beacon block implements the entire beacon protocol [5], including the synchronization methods. The DRP and PCA blocks represent the corresponding protocol implementations. The WiMedia interference block models channel interference for a typical home wireless scenario.

The MAC is stacked over the wireless PHY of ns-2 with two-ray-ground radio propagation, and can be configured with the data rates of the WiMedia PHY. The MAC and PHY layers interact with the interference block to estimate errors on every frame reception, which reasonably mimics a WiMedia application scenario. The interference, DRP and PCA blocks are shown for the sake of completeness, and are not required in the context of this paper.

B. Simulation Setup

A simple two node scenario (Node 0 and 1) was used. The nodes were statically placed less than a meter apart, and their starts were staggered (by few ms) to induce a scenario where one node initiates a BP and other node aligns later. For the purpose of this analysis, which is to verify time synchronization, beacon losses due to channel errors are ignored.

This simple network setting is enough to completely verify

synchronization correctness. As we increase the group size, depicting the relative time error corrections of the entire group would become complex. Maintaining simplicity provides greater clarity in explanation.

In ns-2, all nodes are tied to a single simulator clock, and so the clock drift have to be simulated in each node. The present simulation design implements the clock drift behavior explained in section III.A. Every node, when instantiated, would be initialized with a fixed clock inaccuracy (ppm), and every event scheduled incorporates the drift as a function of the time to schedule the event with respect to the simulation clock. A detailed discussion of the clock drift implementation is beyond the scope of this paper.

The simulation was run for a period of 20 seconds. Both nodes execute all the MAC methods explained before. During the run, the BPST instants of both nodes were profiled to verify the efficiency of the synchronization technique.

C. Simulation Results

Measurements from the entire simulation period are shown in figures 7 and 8. The figures show an average value of six runs to account for the uncertainty in individual measurements.

$Relative\ BPST\ error = (Simulation\ time\ at\ which\ 0\ begins\ a\ SF - Simulation\ time\ at\ which\ 1\ begins\ a\ SF)^8$

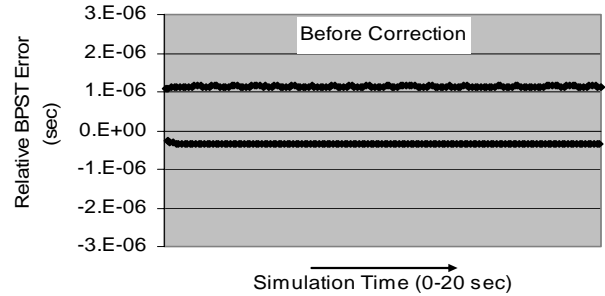


Figure 7. Performance before Correction

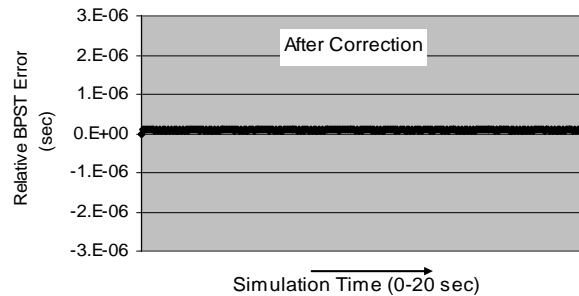


Figure 8. Performance after Correction

Theoretically, time-synchronized nodes should produce a relative BPST error of zero.

The error is plotted for every SF as the simulation runs from 0-20 sec. Since the logical SF structure repeats approximately every 65.536 ms (including μs corrections), the values are excessive to be numbered along the X-axis. Y-axis is scaled

⁸ The simulator clock can serve as the absolute time reference. For time synchronization, 0 and 1 should start every SF at common simulation instants.

at 1 μ s interval, which is the resolution required by the standard.

These graphs show that synchronization is achieved only after incorporating the new-device alignment correction algorithm. The average error is zero after the correction, as compared to a continual error before the correction.

To show the precise timing behavior, figure 9 shows only the first 20 SFs after the new node initiation. The results are shown with 90% confidence intervals (CI) [8], which represent the error margin for the simulation mean value.

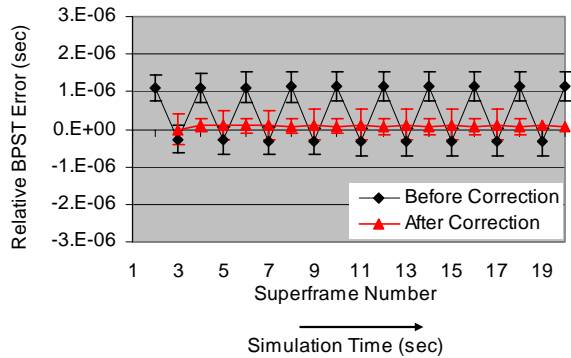


Figure 9. Clear Depiction of Timing Behavior

Before correction: The error constantly oscillates. As stated in IV.A, the new device receives a beacon in the 1st SF of the existing device and erroneously starts the 2nd SF, after the time of instantiation. This initial misalignment never leads to a steady state. When the difference (error) is positive, node 0 is slower, and when negative, node 1 is slower, implying that both nodes alternately over-compensate for each other and never converge. In most SFs, the CI does not encompass the theoretical requirement of zero error, which states with 90% confidence that the nodes never synchronize.

After correction: Nodes converge to a common BPST and the mean error is zero at μ s resolution. The first two SFs' error is not shown since the new-device alignment procedure consumes two SFs, and only after two established SFs from instantiation the new device would have a logical notion of BPST. The graph states, with 90% confidence, that the nodes are aligned.

3-node scenario

Average BPST error measurements from six simulations for a three node scenario, with the correction algorithm, are shown in figure 10, for verifying that this synchronization procedure extends to multiple nodes. The nodes' start times are staggered. From the graph, at μ s resolution;

- The BPST error between 1 and 2 is 0 (red trend line)
- The BPST error between 0 and 1 is 0 (black trend line)
- Implies that all three nodes are aligned

Holding one device in the group as reference is enough for the new devices to synchronize with the entire group, since the reference node would always maintain synchronization with the existing group. So, *by induction*, this new-device BPST correction algorithm (section IV.B) will extend to any BG size with any number of new devices instantiating.

The synchronization results for more than three nodes are not shown, since they provide no extra information, and also the number of trend lines decrease the clarity of results.

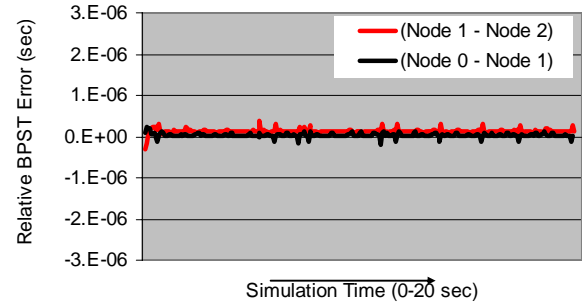


Figure 10. 3-Node Synchronization, after Correction

VI. CONCLUSION

This paper presented a method of time synchronization of devices in a standard for home wireless networks. We provided a critical correction to the current WiMedia [3, 5] synchronization method that enables true time-alignment of new devices. Simulation results showed that if the initial error is not corrected device synchronization may never be achieved, establishing a synchronization error floor for the WiMedia architecture. Although this inaccuracy floor is of the same magnitude as the required clock accuracy, applications requiring tight synchronization (such as high fidelity audio) would perform poorly. Removing this synchronization error floor widens the application domain of the standard, hence substantiating the significance of our new-device alignment method.

With this correction integrated with the WiMedia coordination procedure, time synchronization will be precisely achieved and maintained.

ACKNOWLEDGMENT

We thank Andy Rangekar for his initial ns2 simulation of the WiMedia MAC, proof reading, improvements to the clarity of the paper, challenging questions and constructive discussion.

REFERENCES

- [1] R. Aiello and A. Batra, *Ultra Wideband Systems - Technologies and Applications*, Elsevier, June 2006.
- [2] C. K. Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, Prentice Hall, 2001.
- [3] WiMedia Alliance. <http://www.wimedia.org>
- [4] Multiband OFDM Physical Layer Specification, Release 1.1 July 2005, WiMedia Alliance Inc.
- [5] Distributed Medium Access Control (MAC) for Wireless Networks, Release 1.0 December 2005, WiMedia Alliance Inc.
- [6] Mangold, S. and Choi, S. and May, P. and Klein, O. and Hiertz, G. and Stibor, L. "IEEE 802.11e Wireless LAN for Quality of Service," in Proc. European Wireless '02, Florence, Italy, February 2002
- [7] The Network Simulator. <http://www.isi.edu/nsnam/ns/>
- [8] R. K. Jain, *The Art of Computer Performance Analysis*, John Wiley and Sons, 1992.